ForteTM for JavaTM (Community Edition) QuickStart Guide





Sun Microsystems, Inc. 901 San Antonio Road

Palo Alto, CA 94303 USA 650 960-1300 fax 650 969-9131 Part No.: 806-4569-10 Revision 01, February 2000

Forte for Java (Community Edition) QuickStart Guide

Copyright 2000 Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, California 94303 U.S.A. All rights reserved.

This product or documentation is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or documentation may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Sun, Sun Microsystems, the Sun logo, Forte, Java, JavaBeans, and NetBeans are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

U.S. Government approval required when exporting the product.

DOCUMENTATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, ANY KIND OF IMPLIED OR EXPRESS WARRANTY OF NON-INFRINGEMENT OR THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Copyright 2000 Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, Californie 94303 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Sun, Sun Microsystems, le logo Sun, Forte, Java, JavaBeans, et NetBeans sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

LA DOCUMENTATION EST FOURNIE "EN L'ETAT" ET TOUTES AUTRES CONDITIONS, DÉCLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES DANS LA MESURE AUTORISÉE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE À LA QUALITÉ MARCHANDE, À L'APTITUDE À UNE UTILISATION PARTICULÈRE OU À L'ABSENCE DE CONTREFAÇON.





Contents

Getting Started With Forte for Java 1

Looking at the Forte for Java Environment 1 Editing Workspace 3 GUI Editing Workspace 4 Browsing Workspace 5 Running Workspace 6 Debugging Workspace 7 Working With Forte for Java 8 Creating a Container 9 Choosing a Layout Manager 11 Adding a Label and Setting Its Properties 13 Adding a Button 14 Setting Up the Button to Switch Color 15 Adding Event Handler Code 16 Compiling and Running Your Program 17 Viewing the Class Hierarchy 18 Debugging the Code 19 Taking Best Advantage of Forte for Java 22 Using Templates 22 Using Projects 23 Using Modules 25 Customizing Your Work Environment 25 Getting More Information 26

Glossary 27

Index 31

iv Contents

Getting Started With Forte for Java

Forte[™] for Java[™], Community Edition, provides a complete set of tools, integrated into a single environment, for developing cross-platform applications and applets written in the Java programming language. These tools enable you to edit, compile, debug, browse, and deploy Java programs.

In addition, Forte for Java makes it easy for you to design a graphical user interface (GUI). You can choose components (such as windows, dialog boxes, and buttons) in the component palette and place them in the Form Editor, where you can lay them out. All changes made to the graphical interface (for example, adding a button) are automatically reflected in the source code.

Forte for Java also makes it easy for you to create connections between components. For example, you can create a connection between a dialog box and a button so that the dialog box opens when the user clicks the button. Forte for Java automatically generates the Java code for the action you specify.

Looking at the Forte for Java Environment

When you first start Forte for Java, several windows appear, as shown in FIGURE 1. The *main window* provides access to windows, menus, and toolbars that you can use to develop a Java applet or application. Of particular interest is the *component palette*, where you can access components for building your GUI.

The *Explorer window* provides a hierarchical view of the packages, objects, and files in the Forte for Java environment. The Explorer window is a good starting point for working with the different parts of your application.

The Properties window enables you to view and edit the properties of your components. For example, you can use this window to assign text to a button and to change its color, size, and font.

The Tip of the Day window provides information on how to use the features in Forte for Java more effectively. If you feel comfortable using the environment, you can turn off the display of this window at startup.

The Forte for Java environment includes five workspaces to help you work more efficiently. Each *workspace* consists of a set of windows geared toward a specific task. These tasks are editing, GUI editing, browsing, running, and debugging. You can switch from one workspace to another by clicking its tab in the main window.

FIGURE 1 Forte for Java at Start-up

| | | | Main window |
|---------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------|
| - Forte for lava | Community Edition v. 1.0 | | 1+ 14 |
| File Edit View Project Suild De | bug Tools Window Help | | |
| 0,56 c c d x b c s c c | | AT Sang Seing (Other) Dear | na Layouta Bordera |
| | | | |
| Editing Gui Editing Browning Running Debugging | | | |
| Explorer [Filesystems] · Filesystems Filesystems Matrixalesport/home/gilfforte/Development | Workspace tabs | Component | palette |
| • | Explorer window | | |
| Elesystems | Tip of the Did you know ? Forte far Java, Community Edition modules. You can begin looking right awayjust select Update Help menu. | n, is extensible by plug-in for new modules to add Center from the | Tip of the Day window |
| Project Default 👰 Javadoc 💀 Runtime | E Show Tips on Startup | 1 | |
| Properties Window [No Propertie] | | Next Tip Close | |
| ATRI Propertiese | Properties window | | |

Your current workspace does not constrain the windows you can have open. You can use the View menu on the main window to open any window at any time. When you exit Forte for Java, it saves the state of each workspace. The next time you launch the program, the windows in your workspaces appear exactly as you left them.

Editing Workspace

When you first start Forte for Java, you are in the editing workspace, which contains the Explorer and Properties windows. When you open a file, the *Editor* appears, which is the tool for editing Java, HTML, and plain text files. In the Editor, the source code is syntactically colored—default keywords, for example, are in blue. The Editor also supports *dynamic code completion*—you can type the first few characters of an expression and then view a list of classes, methods, and variables that can be used to complete the expression.The following figure shows the Explorer window, Properties window, and Editor.

FIGURE 2 Editing Workspace



GUI Editing Workspace

The GUI editing workspace is the main area in which you develop a graphical user interface. This workspace includes the Explorer and Editor (as in the editing workspace) plus two additional windows. The *Component Inspector* enables you to view the components in your application and set their properties. The *Form Editor window* is the primary area for creating and modifying a graphical interface. Code generated by the Form Editor appears with a shaded background in the source Editor and can not be edited manually. If you open the Form Editor window in another workspace, Forte for Java automatically switches to the GUI editing workspace. The following figure shows the Component Inspector, Form Editor, and Editor.

FIGURE 3 Component Inspector, Form Editor, and Editor



Form Editor

Browsing Workspace

The browsing workspace includes an *Object Browser* and a Properties window, as shown in the following figure. In the Object Browser, you can view the hierarchy of packages, objects (classes and interfaces), and members (methods, variables, and constructors) in your program. From the Object Browser, you can open the source code of your application by double-clicking a name in either the Objects or Members pane. The Properties window enables you to view and edit the properties of the object selected in the Object Browser.

FIGURE 4 Browsing Workspace



Running Workspace

6

When you run your program, Forte for Java automatically switches to the running workspace, as shown in the following figure. If there are no execution errors, Forte for Java launches your application so that you can test it. If there are execution errors, Forte for Java displays them in the Output window.





Debugging Workspace

The debugging workspace includes the Debugger window and the Output window. The Debugger window has tabbed panes for setting breakpoints, monitoring threads, and watching the value of variables. The Output window displays messages from the debugger. If you have a file open, the debugging workspace also includes the Editor, which highlights breakpoints in magenta. The following figure shows the debugging workspace.

| Debugger window | | |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| - Editor [ClockFrame] | Debugger Window [Breakpoints] | |
| <pre>stmrSecondsOnTime (evt); i i i i i i i i i i i i i i i i i i i</pre> | WindowAdapter .event.@indexEv | |
| <pre>getContentPane .add jlblCurrentTime getContentPane .add jlblCurrentTime getContentPane .add jlblCurrentTime getContentPane .add .ad</pre> | Breakpoints pane Breakpoint highlighted in Editor Messages from the debugger | |
| ClockFlame_BubbleSortAlgorithm_ColorSwitch * | | |
| Gutput III | Thread created: AWT-EventQueue-0 Thread created: Suffoolkit.FoetEventQueue-0 Thread death event: main Ereakpoint reached at line 54 in class tutorial.clcv 4 | |

FIGURE 6 Debugging Workspace

Working With Forte for Java

8

This section guides you through the process of creating a Java application. You'll build a simple program that enables users to switch the color of a panel from light gray to medium gray to black.

Follow these main steps to create an application:

- Create a container from a template and place it in a project.
- Add components to your container, edit the component properties, and create connections between the components.
- Edit the Java source code.
- Compile and run the application.
- View the class hierarchy, if desired.
- Debug the application, if needed.

This "quick start" tutorial takes less than an hour to complete.

Note – Graphics in this document show Forte for Java in the Java look and feel, which is the default configuration.

Creating a Container

You'll begin working in the main window, where you will open the Template chooser and create a top-level *container*. This container will hold the other components (button and label) in your application. You'll create the container using the JFrame component and place it in the colorswitch package. (A *package* is a group of related Java classes and interfaces.)

- **1. From the File menu, choose New From Template.** The Template chooser is displayed.
- 2. In the Template chooser, expand Swing Forms, then select JFrame. A description of the JFrame component appears in the pane on the right, as shown in the following figure.

| Templates | — Template descrip |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| New From Temp | late – Template Chooser |
| Salact a templata | Templete Description |
| Templetes AWT Forms Classes C | Using this template you can design a new JFC (Swing) Frame. Frames are typically used as standalone top-level windows as the main user interface to the application. Most Swing applications are built starting from this form. |
| < Previous Next P | Philip Cancel Help |
| JFrame | 1 |
| Swing Forms | |

FIGURE 7 Template Chooser

3. Click Next.

The Target Location dialog box is displayed.

4. In the Target location dialog box, type ColorSwitch in the Name field and colorswitch in the Package field.

FIGURE 8 Template Location Dialog Box

| | New From Template – Target Location | |
|-----------------------------------------|--------------------------------------------------------------|----|
| ColorSwitch | • | |
| lect Package | | |
| anne select a package | for the newly created object or enter the package name below | a. |
| File systems | | |
| metrice/exportitive | a melgil/forte/Development | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| eclarge : colormettch | • | |
| eckang e) <mark>calamentich</mark> | | |

5. Click Finish.

A Question alert box asks if you want to put the ColorSwitch object in the current project. A project organizes the files in your application into a single group, which makes them easier to find and work on.

6. Click Yes.

The hierarchy in the Explorer window is expanded to show the colorswitch package. In addition, the Component Inspector, Form Editor, and Editor are displayed in the GUI editing workspace. The Component Inspector provides a visual representation of the components in your application and enables you to edit their properties. The Form Editor is the area into which you will add components for this container. The Editor shows the Java source code for the JFrame component. You can type in the white areas of the Editor only.

The following figure shows the colorswitch application in the GUI editing workspace.





Choosing a Layout Manager

A *layout manager* assists you in determining the size and position of the components within the container. Each container type has a default layout manager. BorderLayout is the default for a JFrame component. BorderLayout divides the container into five sections (north, south, east, west, and center). You will switch to GridLayout, which creates sections equal in size and displays them in the requested numbers of rows and columns.

1. In the main window, click the Layouts tab in the component palette.

The Layouts pane provides access to the layout managers in the *Java Foundation Classes* (JFC). When you move the pointer over a button in the toolbar, a tool tip displays the name of the layout manager.

FIGURE 10 Tool Tip for GridLayout in Layouts Pane



- 2. In the Layouts pane, click the GridLayout button.
- **3. Click anywhere in the Form Editor window.** The Form Editor changes to display a grid three columns wide by two rows deep.
- 4. In the Component Inspector window, select GridLayout. The properties of this layout manager appear in the Properties pane of the Component Inspector.
 - FIGURE 11 Component Inspector for JFrame



5. In the Columns property, delete 3, then type 1 and press Enter (or Return). The Form Editor now displays a grid with one column and two rows, as shown in the following figure.

FIGURE 12 Form Editor With GridLayout

| _ | Form [ColorSwitch] | |
|---|--------------------|--|
| | | |
| | | |
| | | |

Adding a Label and Setting Its Properties

Your application uses a blank label with an opaque background to display the colors. You need to add the label and set its properties.

1. In the component palette, click the Swing tab.

The Swing pane provides access to the user interface components in the Java Foundation Classes (JFC).

2. In the Swing pane, click the JLabel button.

FIGURE 13 JLabel in Swing Pane

3. Click anywhere in the Form Editor window.

A label with the text jLabel1 is displayed in the Form Editor. In the Component Inspector, a node named jLabel1 [JLabel] is displayed (and highlighted) in the tree view and the properties for the component are displayed in the Properties pane. Source code for the label is displayed in the Editor.

- 4. In the Properties pane, scroll to the opaque property, click its value, and choose True from the combo box.
- 5. In the text property field, delete jLabel1 (leaving the value blank) and press Enter (or Return).

The Form Editor will again look like FIGURE 12.

Adding a Button

The user interacts with the Color Switch application by clicking a button, which you will now add.

1. In the Swing pane, click JButton.

FIGURE 14 JButton in Swing Pane



- **2.** Click anywhere in the Form Editor. The button is displayed in the Form Editor, its corresponding node is displayed in the Component Inspector, and its source code is displayed in the Editor.
- 3. In the Properties pane, type Switch Color in the text property field, and press Enter (or Return).
- 4. Click in the font property field, and then click the ... button.
- 5. In the Property Editor dialog box, select Bold for font style and 12 for font size, and click OK.

The Form Editor looks as follows.

FIGURE 15 Form Editor With Label and Button



Setting Up the Button to Switch Color

In this section, you specify an event (a mouse click) to which the button can respond.

- 1. In the Component Inspector window, select the jButton1 node (if it is not already selected).
- 2. Click the Events tab.
- 3. In the Events pane, scroll to the mouseClicked field and click. The value changes from <none> to JButtonlMouseClicked.

FIGURE 16 Component Inspector Showing Events for JButton1



4. Press Enter (or Return).

The listener code jButton1.addMouseListener and event method jButton1MouseClicked()automatically appear in the Editor, as shown in the following figure.

FIGURE 17 Editor Showing Listener Code and Event Method

- Listener code jButton1.addMouseListener

| <pre>41 getCantentPane ().add (jLmbell); 42 43 jButton1.setPest ('writin Color"); 44 jButton1.setPest ('writin Color"); 45 jButton1.setPest (new java.set.event.KouseAdapter 47 public void mouseClicked (java.set.event.KouseAdapter 47 jButtoniKouseClicked (set); 48 j 49 j 49 j 40 j; 51 getCantentPane ().add (jButton1); 53 getCantentPane ().add (jButton1); 54 j 55) 56 private void jButtoniMouseClicked (java.set.event.HouseEvent et 57 private void jButtoniMouseClicked (java.set.event.HouseEvent et 58 // Add your headling code here: 59] 60 /** Exif the Application =/ 61 private void emitForm(jeve.set.event.WindowEvent evt) 1 63 System.exit (0); 64] 65 /** fair the commend line arguments 66 /** fource args the commend line arguments 67 * fource args the commend line arguments 68 /** fair INS</pre> | + | | | + |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|-----------|-----|-----|
| <pre>sti getCantentFine ().add ()Label()) jButton1.setFest ('writin Color"); jButton1.setFest ('writin Color"); jButton1.addMouseListener (new java.set.event.KouseAdapter public void mouseClicked (java.set.event.MouseEvent evt)</pre> | | | 5 | 410 |
| <pre>is jButtonl.setText ("Mitch Color"); jButtonl.setText ("Mitch Color"); jButtonl.sddMouseListener (new java.set.event.MouseAdapter public void mouseClicked (java.set.event.MouseEvent evt) jButtonlMouseClicked (svt); } getContentFane ().add (jButtonl); } getContentFane ().add (jButtonl); } getContentFane ().add (jButtonl); } // Add your handling code hare; } // Add your handling code hare; } // System.exit (0); } //** Exif the Application */ private woid emitForm(jeve.set.event.WindowEvent evt) 1 System.exit (0); } /** /** Garae args the commend like arguments //** /** INE /** /** /** /** /** /** /** /** /** /*</pre> | | Q. | | |
| <pre>junction.setFont (new java.set.Font ("Dislog", 1, 12)); jButtonl.addMouseListener (new java.set.event.KouseEdapter public void mouseClicked (java.set.event.MouseEvent evt)</pre> | | | | 40 |
| <pre>jButtonl.addMouseListener (new java.met.event.MouseEvent evt) jButtonl.addMouseListener (new java.met.event.MouseEvent evt) jButtonlMouseClicked (java.met.event.MouseEvent evt) jButtonlMouseClicked (evt); getContentFane ().add (jButtonl); getContentFane ().add (jButtonl); private void jButtonlMouseClicked (java.met.event.MouseEvent evt) // Add your hendling code here: // Add your hendling code here: // Add your hendling code here: // Add your hendling code here: // Add your hendling code here: // Add your hendling code here: // Add your hendling code here: // Add your her</pre> | | | | 1 |
| <pre>public void mouseClicked (java.avt.event.MouseEvent evt) jButtoniMouseClicked (evt); jButtoniMouseClicked (evt); j j getContentPane ().add (jButtoni); j getContentPane ().add (jButtoni); // Add your handling code hare: // Add your handling code hare: j // Add your handling code hare: j for private void (ButtoniMouseClicked (java.ast.event.HouseEvent + // Add your handling code hare: j // Add your handling code hare: // Add your hardling co</pre> | Nonse Minter | 10 | | 12 |
| <pre>jButteniKouseClicked (evt); jButteniKouseClicked (evt); getContentFane ().add (jButteni); getContentFane ().add (jButteni); // Add your handling code here; // Add your handling code here; // Add your handling code here; // Add your handling code here; // Add your handling code here; // Add your handling code here; // Add your handling code here; // Ad</pre> | unsEvent syt) (| | | 10 |
| <pre>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</pre> | SERETRUS STAT 1 | | 2 | 47 |
| <pre>setContentPane ().add (jButtonl); setContentPane ().add (jButtonl); s</pre> | | | | 4.0 |
| <pre>50); 51 52 getContentPane ().add (jButtoni); 53 getContentPane ().add (jButtoni); 54] 55] 56 // Add your handling code hare: 57 // Add your handling code hare: 58 // Add your handling code hare: 59 // Add your handling code hare: 59 // Add your handling code hare: 59 // Add your handling code hare: 50 // Add your handling</pre> | | 1.1 | | 4.9 |
| <pre>state state s</pre> | | 30 | | 50 |
| <pre>32 33 34 35 35 36 36 37 private void jButtaniMauseClicked (java.ast.event.HonseEvent + 38 37 28 38 39 39 39 39 39 39 39 39 39 39 39 39 39</pre> | | | | 51 |
| <pre>53 getContentFane ().add (jButtonl); 54 ; 55 ; 56 ; 57 private void jButtoniMouseClicked (java.ast.event.HouseBvent + 56 // Add your handling code hare; 57 ; 58 // Add your handling code hare; 59 ; 59 ; 50 /** Exit the Application */ 59 ; 59 ; 50 ; 50 ; 50 ; 50 ; 50 ; 50 ; 50 ; 50</pre> | | | 2 | 5.2 |
| <pre>54 55 57 private void jButtaniMauseClicked (java.ast.event.HonseEvent + 58 77 Add your handling code hare: 59 67 77** Exit the Application * 7 67 67 67 67 7** Exit the Application * 7 68 7** 68 7** 69 7** 69 7** 69 7** 60 1 60 50 1 60 50 1 60 50 1 60 50 1 60 50 1 60 50 50 1 60 50 50 1 60 50 50 50 50 50 50 50 50 50 50 50 50 50</pre> | | ge | 1 | 50 |
| <pre>55) 56 56 56 56 56 56 56 56 56 56 56 56 56</pre> | | | 1 | 54 |
| <pre>34 57 private wold jButtanlMauseClicked (java.ast.event.HouseEvent - 56 // Add your handling code hare: 57 3 58 59 /** Exit the Application */ 59 private wold emitForm(java.ast.event.WindowEvent evt) 1 59 system.emit (0); 54 1 55 sil 102 59 sil 102 59 sil 102 59 sil 102</pre> | | 3 | 5 | 55 |
| <pre>57 private void jButtoniMouseClicked (java.ast.event.HouseEvent + 55 // Add your handling code hare: 59] 50] 50 [/** Exit the Application */ 50 private void exitForm(java.ast.event.WindowEvent evt) 1 51 System.exit (0); 53] 54] 55] 50 1 102 50 1 103 50 1 104 50 1 50 1 50 1 105 50 1 105 50 1 105 50 1 105 50 1 105 50 1 105 50 1 50 1</pre> | | | e | 5.6 |
| <pre>54 // Add your handling code hare: 59] 60 61 /** Exit the Application */ 62 private wold exitForm(java.art.event.WindowEvent evt) 1 63 System.exit (0); 64] 65] 66 /** 67 * @person args the commend line arguments 50:1 [102]</pre> | t.HouseEvent evt | privat | 7 p | 57 |
| <pre>19 } 19 } 10 /** Exit the Application */ 10 private wold exitForm(java.set.event.WindowEvent evt) 1 10 System.exit (D); 10 1 10 10 10 10 10 10 10 10 10 10 10 10 10</pre> | | 11 Add | 5 / | 50 |
| <pre>00 61 /** Init the Application */ 62 private void emitForm(java.avt.event.WindowEvent evt) 1 63 System.exit (0); 64 1 65 c 66 /** 67 * dparam args the command line arguments 59:1 INS</pre> | | - 1 | 2 | 5.9 |
| <pre>61 /** Isit the Application * 62 private void esitForm(java.avt.event.WindowEvent evt) 1 63 System.exit (0); 64 1 65 66 /** 67 * dparam args the command like arguments 59:1 INS</pre> | | | 2 | 60 |
| <pre>50 1 INS</pre> | | 140 | 5 | 61. |
| <pre>63 System.exit (0); 64 1 65 66 /** 87 * dperson args the command line arguments 50:1 INS</pre> | : evt) (| prix | z | 62 |
| 04 3 05 06 /** 07 * Aperan args the command line arguments 50:1 [102] | | Sy | 3 | 63 |
| 59:1 INC | | 1 | • | 04 |
| 00 /** 87 * Aperan args the command line arguments 50:1 [102] | | | 5 | 65 |
| 50:11 INS | | 1 | 0 | 66 |
| 50:1 [105] | | * 8P | 7 | 67 |
| 2611 1000 | | | | - |
| | | WIL | 29 | 3 |
| RubbleSontAlgorithm ClockFiame ColorSwitch * | | oleSortA) | bbl | Bub |

Event method jButton1MouseClicked()

Adding Event Handler Code

Now that Forte for Java has created the jButtonlMouseClicked() method, you can add custom code for handling this event. You want a click of the button to change the color of the label.

1. In the Editor, after the comment //End of variables declaration (line 77), declare a new variable:

private java.awt.Color currentColor = java.awt.Color.lightGray;

(To use the dynamic code completion feature in the Editor, type the first few characters, choose the completed term from the list of classes, methods, and variables that the Editor displays, and press Return.)

2. After the lines 57 and 58:

private void jButton1MouseClicked (java.awt.event.MouseEvent evt) {
 //Add your handling code here:

type the following:

```
if (currentColor == java.awt.Color.lightGray)
    currentColor = java.awt.Color.gray;
else if (currentColor == java.awt.Color.gray)
    currentColor = java.awt.Color.black;
else
    currentColor = java.awt.Color.lightGray;
jLabel1.setBackground (currentColor);
```

3. From the File menu, choose Save.

Compiling and Running Your Program

Now that you've created the user interface and the connections between the components, you can compile and run your application.

1. From the Project menu, choose Compile Project.

If the build is successful, you'll see "Finished Project Default" in the status bar in the main window. If there are problems with the build, you'll see error messages in the Output window. You can double-click an error message to jump to the line in the source code where the error occurred.

Once the build is successful, you can run the application.

2. From the Project menu, choose Execute Project.

If there are no execution errors, Forte for Java switches to the running workspace and the Execution View window, the Output window, and the Select Main Class dialog box appear.

3. In the Select Main Class dialog box, select ColorSwitch, then click OK. The colorswitch application is displayed, as shown in the following figure.

FIGURE 18 Running Workspace With colorswitch application

| colorswitch | Execution View | |
|---------------|-----------------------------------------------|---|
| AWTa | Execution View Colorswitch.GalarSwitch | |
| Output window | Output Window [colorswitch.ColorSwitch – I/O] | |
| Program input | |] |

- 4. In the application, click the button three times to check that the label changes from light gray to medium gray to black.
- 5. Close the application.

Viewing the Class Hierarchy

Forte for Java has an Object Browser that enables you to view the classes, methods, and data items in your program.

1. In the main window, click the Browsing tab.

The Object Browser window is divided into the Packages, Objects, and Members panes, as shown in the following figure.

| colorswitch package | ColorSwitch class | | jBut | tonl |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|-------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|-----------------|
| | Object Browser | | | |
| Packages | Objects | | | Menten |
| A1 + 4 0 | 2 7 6 C E ~ G | 0 | mv | 1 2 2 2 2 3 |
| Anetlickeviexport2/home/gail/forte/Devel colonwetch examples examples.advanced examples.bubblesort examples.bubblesort examples.bubblesort examples.bubblesort tutorial tutorial tutorial tutorial clock part1 tutorial clock part2 tutorial clock part3 tutorial clockpart3 tutorial clockpart3 tutorial clockpart3 tutorial clockpart4 tutorial clock part3 tutorial clockpart3 tutorial clockpart4 tutorial clockpart4 tutorial clockpart4 tutorial clockpart4 tutorial clockpart4 tutorial clockpart4 tutorial tutorial clockpart4 tutorial tuto | 🚰 ColorSettch | An cu de ja An ju An ci An ci An ci An ci An ci An ci | anntGalar utton 1 abeil olo:Switch ofParm itComponen utton 1Mous ain | 15 e Clicked |

FIGURE 19 Colorswitch Application in the Object Browser

- 2. In the Packages pane, select the colorswitch package. The Objects pane now shows the ColorSwitch class.
- **3.** Select the ColorSwitch class. The Members pane shows each method and data member contained in the ColorSwitch class.
- 4. Select JButton1.

The button's properties appear in the Properties window.

5. Double-click the main method.

The GUI editing workspace is displayed. In the Editor, the cursor appears at the beginning of the line with the main method.

Debugging the Code

Using the debugger, you can locate and correct bugs in your program. The following steps introduce you to debugging in Forte for Java.

- 1. In the GUI editing workspace, click line 27 in the Editor, which contains the initComponents method.
- **2. From the Debug menu, choose Add/Remove Breakpoint.** The line is highlighted in blue to indicate a breakpoint was set.

3. From the Debug menu, choose Start Debugging.

The program runs until it reaches the breakpoint. Forte for Java switches to the debugging workspace, which displays:

- The breakpoint line highlighted in magenta in the Editor
- Information on the breakpoint in the Debugger window
- Messages from the debugger in the Output window

FIGURE 20 Colorswitch Application Stopped at Breakpoint

| | Editor [ColorSwitch] | | |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|
| 10 17 10 19 | /** Creates new Form ColorSwitch */ public ColorSwitch() { initGomponents (); pack (); | | Debugger window with breakpoints pane displayed |
| 20 | 4. (J. 1997) | Debugger Window | / [Breakpoints] |
| 22 23 24 25 26 | /** This method is called from within the s * initialize the form. * WARNING: Do NOT modify this code. The co * always regenerated by the TormEditor. */ | Breakpoints | - No Frequencies |
| 27 28 29 30 31 32 32 34 | <pre>private weld initComponents () { Stabell = new javax.swing.Jabel (); Sattonl = new javax.swing.Jautton (); getContentFane ().setEsyout (new java addWindowListener (new java.awt.event.) public wold windowClosing (java.awt. emitForm (evt); 1 </pre> | | |
| 25 | 1 | 🖶 Breakpoints 🛂 Threads 🗔 Wato | heo |
| 27 38 29 40 41 4 Cloc | jLabell,setOpaque (true); getContentPane ().add (jLabell); 7:1 INE WForme BubbkSortAugorithm ColorSwitch | | Output window showing messages from the debugger |
| - | Output Win | dow [Debugger] | · 🗌 |
| | | Thread created: Step hand Thread created: AWT-Event Thread created: SunToolki Thread created: AWT-Notif Erreakpoint reached at lin | ller • • • • • • • • • • • • • • • • • • • |

- Editor with breakpoint highlighted

- 4. From the Debug menu, choose Trace Into. The call to pack in the method ColorSwitch is highlighted in the Editor.
- **5. From the Debug menu, choose Trace Over.** The next method call is highlighted.
- **6. From the Debug menu, choose Trace Out.** The first line in the main method is highlighted.
- 7. In the Editor, click the line with the initComponents method and then choose Add/Remove Breakpoint from the Debug menu. The breakpoint is removed.
- **8.** In the Debugger window, click the Threads tab, then expand the system node. You'll see the debugger and other threads in your program.
 - FIGURE 21 Threads Tabbed Pane in Debugger Window



9. In the Debugger window, click the Watches tab. You can use this pane to display the value of variables in your program.

FIGURE 22 Watches Tabbed Pane in Debugger Window



10. From the Debug menu, choose Finish Debugging.

Taking Best Advantage of Forte for Java

The previous tour of Forte for Java introduced you to the main interface and the steps involved in creating an application. Following are some additional features of the environment that enable you to create applets and applications more efficiently.

Using Templates

Templates are a powerful tool in the Forte for Java environment. When you create an object, you use its *template*, which determines the initial appearance and behavior of the object. Templates can reduce the amount of time and effort involved in creating your application.

Java components, such as Swing and AWT containers, are provided as standard templates. Forte for Java also provides templates for applets, classes, dialog boxes, HTML files, text files, and bookmarks. To access the Template chooser (shown in the following figure), choose New From Template from the File menu.

FIGURE 23 Template Chooser

| New From Template – Template Chooser | | | | |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|--|
| Select a template Templates AWT Forms Classes Classes Sample Porms Sample Porms Select template Select template Sele | Templete Description Using this templete you can design a new Serviet class. Serviet is a server-side Java class which runs within a web server. | | | |
| < Previous Next P | Finah Garcel Help | | | |

Using Projects

A *project* enables you to organize the files required to produce an applet or an application. When you organize your files into a project, you can operate on them as a whole. For example, when you compile a project, you compile all of the Java source files in it.

You create and manage projects using the Project menu in the main window. To view the files in the current project, use the Project tab in the Explorer window, as shown in the following figure.

FIGURE 24 Project Tab in Explorer Window

| - Explorer [Project Default] | |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| Project Default | |
| S GalarSwitch (->) | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | Project tak |
| | |
| T Flesystems O Project Default (2) Javados 🖙 Buntime | |
| and the second s | |

For each project, you can specify a set of attributes, such as which compiler and debugger types to use. You set attributes in the Project Settings dialog box (shown in the following figure), which is available from the Projects menu.

FIGURE 25 Project Settings Window

| - Proj | ect Settings | + | |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|----------------------|--|
| Project Settings Privacystems Settings Privacystems Settings Project Options | ¥ 19 X 8= 1 | | |
| | Debug | Tiue | |
| | Deprecation | Paine | |
| | Encoding | null | |
| | Identifying Name | Internal Compilation | |
| | Optimize | Fake | |
| | Properties | | |

Using Modules

Forte for Java is built entirely from modules, independent pieces of software that are compiled separately. Even features central to the environment, such as the Editor, Debugger, and Form Editor, are modules. You can expand this modular structure with plug-in extensions from Sun Microsystems or third parties. To view the modules currently installed in your environment, choose Global Options from the Tools menu. When the Global Options window opens, expand the Modules node, as shown in the following figure.

Global Options 🔁 Global Options × 1 2 花 100 Actions - ET Toolbars Enabled Tiue 8- 🖪 Startup HTML Marris 0.9.3 - B. Menu Expanded 🗢 🗩 Object Types 0.9.3 modules 🛛 🏂 Modules node HTTP Server A los Browser A Sourceless Java Classes PL HTML h Inape Text h User's Guide A GVS A Common Utilities Auto Lipciata A Object Browser Debugger Core Properties Expert Apple1

FIGURE 26 Global Options Window With Modules Node Expanded

To install a new module, choose Update Center from the Help menu and follow the directions in the dialog box.

Customizing Your Work Environment

Forte for Java lets you customize your work environment in a number of ways. You can create, delete, and change menus, toolbars, and workspaces. You can also add an object to the component palette and then use that object in your applets and applications. You can look at files in the Editor side-by-side, instead of in separate tabbed panes, and you can change the Editor's keyboard shortcut assignments. You can make these changes (and change other aspects of your environment) in the Global Options window, shown in the preceding figure. You access the Global Options window from the Tools menu.

Getting More Information

The Forte for Java (Community Edition) User's Guide describes both conceptual information and how to use Forte for Java. You can view this document online by opening the Help menu and then choosing Documentation. In addition, you can press F1 in most windows to open the user's guide to information specific to the task you are performing. The following figure shows a page from the user's guide.

FIGURE 27 User's Guide Displayed in Help Window



Forte for Java also includes four online tutorials, which introduce you to additional features of the environment, such as how to build a JavaBeans[™] component architecture. To access the tutorials, open the Help menu, and choose Tutorial.

From the Help menu, you can also open a web browser and access the Forte for Java web site (www.netbeans.com) by choosing Forte for Java Home on the Web. This site provides installation instructions, module updates, and other information.

Glossary

| Abstract Window Toolkit | An API that provides graphical user interfaces for Java programs. The Abstract Window Toolkit (AWT) also provides imaging tools, event-handling methods, layout managers, and data transfer classes. The AWT components are implemented using native-platform versions of the components and have largely been replaced by the Swing components, which have a pluggable look and feel. |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| applet | A program, written in the Java programming language, that runs in a web browser. |
| application | A standalone software program that enables the user to perform a specific task, such as database management. |
| class | A group of attributes and methods that define the implementation of a particular type of object. |
| component | An object that is identified by its properties, operations, and relationships. For example, a button is a visual component whose properties include size and foreground and background color. |
| Component Inspector | A window in which you can view both the visual (such as a button) and non-visual (such as a layout manager) components in your application. From the Component Inspector, you can modify a component's properties and specify its events. |
| component palette | A collection of toolbars that provides easy access to frequently used components, including AWT and Swing components. You can create your user interface by clicking a component in the component palette and then clicking in the Form Editor. |
| container | A component that contains other components. Windows and dialog boxes are examples of top-level containers. Panels, scroll panes, and tabbed panes are examples of intermediate-level containers. |
| dock | To anchor an object, such as a toolbar, to the edge of the window or pane to which it applies. |
| dynamic code completion | The automatic completion of an expression that you are typing in the Editor window. To use dynamic code completion, type the first few characters of the expression, and then press CTRL+SPACE. A list of of classes, methods, and variables that can be used to complete the expression is displayed. |

| Editor | A tool for editing Java, HTML, and plain text files as well as files specified by modules. Source code in the Editor window is automatically updated and generated as you work in the Form Editor window and Component Inspector. The generated source code is indicated by a shaded background and cannot be edited in the Editor window. |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| event | An action to which an object can respond. Most events are initiated by a user action, such as a click, key press, or mouse movement. |
| event handler | A method that is called when an event is triggered on a component. |
| Explorer window | A window that provides a unified view of all objects and files in the Forte for Java environment. The Explorer window is a good starting point for working with your application, including organizing files, editing object properties, and creating component connections. |
| Form Editor window | An area for creating and modifying a graphical interface. You can select a component (such as a panel, scroll bar, or menu) in the component palette and add it to your graphical interface by clicking in the Form Editor window. |
| Java Foundation Classes | An extension to the Abstract Window Toolkit (AWT) that provides the Swing classes, a collection of graphical user interface components with a pluggable look and feel. The Java Foundation Classes (JFC) also provide the Java Accessibility API, which can be used to create applications that interact with assistive technologies. For the Java 2 platform, the JFC also includes the Java 2D API (for 2D graphics and imaging) and drag and drop. |
| layout manager | A property of a container component that controls the size and location of components within the container. A layout manager ensures that the container can adjust to resizing and to differences between systems, such as different font sizes. The Java platform supplies six layout managers: BorderLayout, BoxLayout, CardLayout, FlowLayout, GridBagLayout, and GridLayout. |
| main window | A window that acts as the control center for Forte for Java. The main window contains menus, toolbars, and a component palette for developing Java applets and applications. From the main window you can access the five workspaces in Forte for Java. |
| method | A procedure that belongs to a class and that can be applied to a specific object or the class itself. |

- moduleAn independent piece of software that is part of a larger program but
is usually compiled separately. Modules are implemented in such a
way that you can change one module without affecting the other
- **Object Browser** A three-pane window in which you can view the hierarchy of packages, objects, and members in your application. Like the Explorer window, you can use the Object Browser as a base for many tasks in the development of your application. For example, you can open a source file and add new packages, objects, and members from the Object Browser.

modules in the program.

- package A collection of Java classes and interfaces, grouped in a single entity.
- **project** A collection of files that make up an applet or application. The files in a project can be operated on as a whole.
- **property** An attribute or characteristic of a GUI object that you can set. The properties of an object might define its size, color, and value.
- Swing
componentsA collection of GUI components with a pluggable look and feel so
you can design an application that can have the look and feel of any
OS platform. Swing is part of the Java Foundation Classes and
includes interface elements such as windows, dialog boxes, choosers,
panels, panes, menus, controls, text components, tables, lists, and tree
views.
- templateSoftware code that serves as a guide for creating a component. A
template provides the initial appearance and behavior of the object,
which you can easily change. In the Forte for Java environment,
components such as Swing and AWT containers are provided as
templates.
- workspace A collection of windows with related functions. For example, when you edit your user interface, you use a workspace that displays the Component Inspector, the Form Editor, and the source Editor. When you debug your program, you use a workspace that displays a window for setting breakpoints, monitoring threads, and watching the value of variables.

30 Glossary

Index

A

applets, 22 applications running, 17 steps in creating, 8

В

BorderLayout layout manager, 11 breakpoints, 7, 19–21 browsing source files, 18–19 browsing workspace, 5 building Java programs, 17 buttons, adding to containers, 14

С

code completion in Editor, 3 code generation, automatic, 1 compiling source code, 17 Component Inspector, 4, 10 component palette adding custom components to. 25 overview, 1–2 components adding events to, 15 adding to containers, 13 adding to palette, 25 interactions. 16 properties, 13 containers, 9 customizing the environment, 25

D

debugging source code, 19–22 debugging workspace, 7 documentation, 26

Ε

editing event handler methods, 16 source files, 3 editing workspace, 3 Editor, 3, 10 environment, customizing, 25 errors, locating in source code, 17 event handler methods, 16 events, adding to components, 15 Explorer window, 1

F

Form Editor, 4, 10

G

GridLayout layout manager, 11 GUI editing workspace, 4

Η

help, 26 HTML files, 22

J

Java Foundation Classes, 11, 13 JButton component, 14 JFrame component, 9 JLabel component, 13

κ

keyboard shortcuts, 25

L labels, adding to containers, 13 layout managers, 11–13 Layouts pane, 11

Μ

main window, 1–2 modules, 25

0

Object Browser overview, 5 using, 18–19 online help, 26 Output window, 6, 20

Ρ

packages, 9 projects, 23–24 Properties pane, 12 Properties window, 1 properties, modifying component, 13

R

running an application, 17 running workspace, 6

S

Swing components, 13

Т

templates, 22–23 threads, monitoring, 7, 21 Tip of the Day window, 1 tool tips, 11 top-level containers, 9

۷

variables, watching, 7, 21–22

W

workspaces, 1–7 browsing, 5 debugging, 7 editing, 3 GUI editing, 4 running, 6